Compositionality via Cut-Elimination: Hennessy-Milner Logic for an Arbitrary GSOS

Simpson, A. (1995). Categorical Completeness Results for the Simply-typed Lambda-calculus. In Proceedings of the Second International Conference on Typed Lambda Calculi and Applications. (pp. 414-427). (TLCA '95). London, UK, UK: Springer Verlag. doi: 10.1007/BFb0014068

Compositionality via cut-elimination: Hennessy-Milner logic for an arbitrary GSOS

Alex K. Simpson
LFCS, Department of Computer Science, University of Edinburgh,
JCMB, King's Buildings, Edinburgh, EH9 3JZ, UK.
Email: Alex.Simpson@dcs.ed.ac.uk

Abstract

We present a sequent calculus for proving that processes in a process algebra satisfy assertions in Hennessy-Milner logic. The main novelty lies in the use of the operational semantics to derive introduction rules (on the left and right of sequents) for the different operators of the process calculus. This gives a generic proof system applicable to any process algebra with an operational semantics specified in the GSOS format. We identify the desirable property of compositionality with cut-elimination, and we prove that this holds for a class of sequents. Further, we show that the proof system enjoys good completeness and ω-completeness properties relative to its intended model.

1 Introduction

The provision of proof systems for program logics is an important research goal, as such systems enable one to give formal proofs guaranteeing that programs satisfy required properties. A desirable feature of such proof systems is that they should allow a *compositional* style of proof development. Informally, a proof system is compositional if it builds a proof that a compound program satisfies a compound property out of proofs that the constituent subprograms satisfy relevant subproperties. Compositionality is important for several reasons (see, e.g., [10, 9, 2]), not least because it allows a proof that a program satisfies a property to be built up following the structure of the program and property being verified.

The work in this paper is based on the observation that, in the context of pure first-order logic, the issue of compositionality was addressed long ago by Gentzen in his work on sequent calculus [4]. In sequent calculus, each of the connectives has rules building a conclusion involving a compound formula out of premises involving its immediate subformulae. Only one rule violates the structure-building nature of compositionality: the cut rule. However, cut is eliminable. Thus Gentzen obtained compositionality via cut-elimination.

In this paper we develop a worked example to show how the techniques of sequent calculus can similarly be used to address the issue of compositionality in program logics. Our example is a sequent calculus for showing that processes in any process algebra with an operational semantics specified in the GSOS format [3, 1] satisfy assertions of Hennessy-Milner logic [6]. Such process algebras provide interesting examples because of the well-known difficulties in giving proof rules for parallel operators [9, 10]. The benefit of working with an arbitrary GSOS system is that we obtain a generic proof system applicable to a wide class of process algebras.

In our setting, one is interested in establishing that a process p satisfies a formula A. We therefore build sequents from sets of judgements of the form $p : A$, which are to be read as expressing such properties. Sequents have the standard form $\Gamma \Longrightarrow \Delta$, where Γ and Δ are finite sets of judgements considered as conjoined and disjoined respectively. Our methodology is to give rules for sequents involving the usual style of introduction rule (on the left and right of sequents) both for formulae and for processes.

For the formulae of Hennessy-Milner logic we need such rules both for the propositional connectives and for the modalities. The rules for the former are standard. For the modalities, we give rules which reflect in as direct a way as possible their meanings. For example, in the case of the necessity modality, we have that p satisfies $[a]A$ (where a is some action) if and only if, for every process q such that p can perform a to become q (notation $p \xrightarrow{a} q$), it holds that q satisfies A. In order to translate this in terms of primitive rules it is necessary to have a further judgement form expressing that $p \xrightarrow{a} q$ for processes p and q. Then one

has natural rules:

$$\frac{\Gamma \Longrightarrow p \xrightarrow{a} q, \Delta \quad \Gamma, q{:}A \Longrightarrow \Delta}{\Gamma, p{:}[a]A \Longrightarrow \Delta}$$

$$\frac{\Gamma, p \xrightarrow{a} x \Longrightarrow x{:}A, \Delta}{\Gamma \Longrightarrow p{:}[a]A, \Delta}$$

where, in the second rule, x is a variable (ranging over processes) that does not appear in the concluding sequent of the rule (thus x is an *arbitrary* process to which p can evolve via a). The inclusion of process variables involves allowing judgements to contain open process terms. This increases expressivity: one can state general properties ranging over the set of all processes. This possibility raises the question of ω-completeness (see Section 5).

The rules for processes are derived from the operational semantics of the process algebra, making crucial use of the presence of $p \xrightarrow{a} q$ judgements. Indeed, the right-hand rules are copied directly from the operational semantics. For example, the rules for the CCS prefix and sum operators [8] are:

$$\frac{}{\Gamma \Longrightarrow a.p \xrightarrow{a} p, \Delta}$$

$$\frac{\Gamma \Longrightarrow p \xrightarrow{a} p', \Delta}{\Gamma \Longrightarrow p+q \xrightarrow{a} p', \Delta} \qquad \frac{\Gamma \Longrightarrow q \xrightarrow{a} q', \Delta}{\Gamma \Longrightarrow p+q \xrightarrow{a} q', \Delta}$$

The rules introducing process operators on the left express that $f(p_1, \ldots, p_k) \xrightarrow{a} r$ may only happen if it is derivable via one of the operational rules for f. For example, for the prefix, zero and sum operators of CCS, this property is expressed by the following rules:

$$\frac{\Gamma[p,r] \Longrightarrow \Delta[p,r]}{\Gamma[r,p], a.p \xrightarrow{a} r \Longrightarrow \Delta[r,p]} \qquad \frac{}{\Gamma, a.p \xrightarrow{b} r \Longrightarrow \Delta}\ a \neq b$$

$$\frac{}{\Gamma, 0 \xrightarrow{a} r \Longrightarrow \Delta}$$

$$\frac{\Gamma, p \xrightarrow{a} r \Longrightarrow \Delta \quad \Gamma, q \xrightarrow{a} r \Longrightarrow \Delta}{\Gamma, p+q \xrightarrow{a} r \Longrightarrow \Delta}$$

where we write $\Gamma[p,r]$ for $\Gamma[p/x, q/y]$. (Incidentally, we did not mention any right-hand rules for zero because there are none.) All the above rules are compositional in the sense that a conclusion involving a process $f(p_1, \ldots, p_k)$ is derived from premises mentioning only its arguments $p_1, \ldots, p_k$.

Although we have not given a full definition of the proof system, we can illustrate that, unfortunately, cut is not eliminable. For example, the following derivation just uses the above process rules and cut:

$$\dfrac{\dfrac{\dfrac{}{b.0 \xrightarrow{c} 0 \Longrightarrow}}{c.0 \xrightarrow{c} 0,\ a.b.0 \xrightarrow{a} c.0 \Longrightarrow} \qquad \dfrac{}{\Longrightarrow c.0 \xrightarrow{c} 0}}{a.b.0 \xrightarrow{a} c.0 \Longrightarrow}\ \text{cut}$$

but there is no cut-free derivation of the concluding sequent. The sequents $a.b.0 \xrightarrow{a} x,\ a.c.0 \xrightarrow{a} x \Longrightarrow$ and $a.b.0 + c.d.0 \xrightarrow{a} x,\ a.b.0 + c.d.0 \xrightarrow{c} x \Longrightarrow$ give other examples of the same phenomenon. This failure of cut-elimination does not seem to be a result of the particular formulation of the rules, but rather an unavoidable problem for the particular sequents considered above. As seems reasonable, all the rules are sound (in a sense explained in Section 3) relative to models in which bisimilar processes are identified. So the only way to show the impossibility of $a.p \xrightarrow{a} q$ is to show that p and q are not bisimilar. This involves considering the hereditary behaviour of p and q, and a cut will be required to remove the resulting contradiction.

As the cut rule violates compositionality, its non-eliminability threatens the whole programme we are advocating. Fortunately, it turns out that if one makes certain restrictions to the class of sequents (excluding, amongst others, the sequents above) then cut *is* eliminable (see Section 3). We show this in Section 4 by proving completeness for the cut-free system (on restricted sequents), thus avoiding a syntactic cut-elimination argument. The completeness result is relative to a class of models, corresponding roughly to the class of those transition systems determined by extensions of the process algebra with new operators. Normally, however, one is interested in the process calculus at hand, which forms the "intended" model. In Section 5, we show that, for certain sequents, the system is complete for deriving truth in the intended model, and we give necessary and sufficient conditions for a useful form of ω-completeness to hold.

2 Preliminaries

We use $x, y, z, \ldots$ to range over a countably infinite set of *process variables*. We use $f, g, \ldots$ to range over a set of *operator symbols* each of which has an associated arity ≥ 0. We use $p, q, r, \ldots$ to range over *process terms* built from the operators and variables. We write $r(\vec{x})$ to mean that all the variables of r are contained in the vector of distinct variables $\vec{x}$; in which case, given a vector of process terms, $\vec{p}$, of the same length as $\vec{x}$, we write $r(\vec{p})$ for the process term obtained by the evident

substitution. We write $Vars(p)$ for the set of variables appearing in p. We say that p is *closed* if $Vars(p) = \emptyset$.

The operational semantics is to be specified by a GSOS system. We follow the treatment in [1], whose limitations will be discussed in Section 7. We use $a, b, c, \ldots$ to range over a *finite* set of *actions*. A *GSOS rule* has the form:

$$\frac{\{x_i \xrightarrow{a_{ij}} y_{ij}\}_{1\le j\le m_i}^{1\le i\le k} \quad \{x_i \not\xrightarrow{b_{ij}}\}_{1\le j\le n_i}^{1\le i\le k}}{f(x_1,\ldots,x_k) \xrightarrow{c} r(\vec{x},\vec{y})} \qquad (1)$$

where: all the variables are distinct; $\vec{x}$ and $\vec{y}$ are the vectors of all x_i and y_{ij} variables respectively; $m_i, n_i \geq 0$; and k is the arity of f. We say that f is the *operator* of the rule and c is its *action*. A *GSOS system*, $\mathcal{R}$, is given by a set of GSOS rules containing, for each operator-action pair f, c, only a finite number of rules with operator f and action c. Henceforth, we assume given a fixed GSOS system, $\mathcal{R}$.

Normally the GSOS system is used to determine a labelled transition system between closed processes giving their operational behaviour. We shall be interested in this transition system as one intended model amongst a wider class of models. First, some preliminary definitions. A *labelled transition system* is a structure of the form $T = (|T|, \{\xrightarrow{a}_T\})$ where: $|T|$ is a set (of *states*); and $\xrightarrow{a}_T$ is a binary relation on $|T|$ for each action a. We use $s, t, \ldots$ to range over $|T|$. We write $s \not\xrightarrow{a}_T$ if there does not exist t such that $s \xrightarrow{a}_T t$. We use *bisimilarity* to refer to the relation of strong bisimilarily between two transition systems [8].

A *premodel* is a structure $T = (|T|, \{\xrightarrow{a}_T\}, \{f_T\})$ where: $(|T|, \{\xrightarrow{a}_T\})$ is a labelled transition system; and f_T is a k-ary function on $|T|$ for each operator f of arity k. Given a premodel T, an *environment* is a function from process variables to $|T|$. An environment, γ, induces an evident function mapping each process term p to a state $\gamma(p) \in |T|$.

We say that a premodel T is a *model* if we have that $f_T(s_1,\ldots,s_k) \xrightarrow{c}_T t$ if and only if there exist an environment γ and a rule in $\mathcal{R}$ (of the form in (1) above) such that:

1. $\gamma(x_1) = s_1$ and ... and $\gamma(x_k) = s_k$;
2. for all i, j with $1 \leq i \leq k$ and $1 \leq j \leq m_i$, it holds that $\gamma(x_i) \xrightarrow{a_{ij}}_T \gamma(y_{ij})$;
3. for all i, j where $1 \leq i \leq k$ and $1 \leq j \leq n_i$, it holds that $\gamma(x_i) \not\xrightarrow{b_{ij}}_T$; and
4. $t = \gamma(r(\vec{x},\vec{y}))$.

Of particular interest is the intended model given by the process calculus itself. This model, $\mathcal{T}_{\mathcal{R}}$, is the unique model based on the algebra of closed process terms. The existence and uniqueness of $\mathcal{T}_{\mathcal{R}}$ is one of the fundamental properties of any GSOS system [3]. Our more general class of models includes all quotients of the intended models of *disjoint extensions* of $\mathcal{R}$ (see [1]) by congruence relations contained in bisimilarity.

We use $A, B, C, \ldots$ to range over formulae of Hennessy-Milner logic [6], which are given by the grammar:

$$A \;::=\; \top \mid \neg A \mid A \wedge B \mid \langle a\rangle A.$$

The other connectives and the $[a]$ modality can be defined in the standard ways. Given a labelled transition system, $T = (|T|, \{\xrightarrow{a}_T\})$, the "forcing" relation, $\Vdash_T$, between $|T|$ and formulae is defined as usual: $t \Vdash_T \top$ always holds; $t \Vdash_T \neg A$ if $t \not\Vdash_T A$; $t \Vdash_T A \wedge B$ if both $t \Vdash_T A$ and $t \Vdash_T B$; and $t \Vdash_T \langle a\rangle A$ if there exists t' such that $t \xrightarrow{a}_T t'$ and $t' \Vdash_T A$.

3 The sequent calculus

In this section we present the sequent calculus. As motivated in Section 1, it has different judgement forms: *logical judgements* $p\!:\!A$; and *action judgements* $p \xrightarrow{a} q$. In addition, as the operational semantics allows negative premises, we include *inaction judgements* of the form $p \not\xrightarrow{a}$. We use $J, K, \ldots$ to range over judgements and $\Gamma, \Delta, \ldots$ to range over (possibly infinite) sets of judgements.

Judgements have interpretations in arbitrary premodels. A relation $T \models_\gamma J$ between premodels T, environments γ and judgements J is defined by:

$$\begin{array}{lll} T \models_\gamma p \xrightarrow{a} q & \text{if} & \gamma(p) \xrightarrow{a}_T \gamma(q), \\ T \models_\gamma p \not\xrightarrow{a} & \text{if} & \gamma(p) \not\xrightarrow{a}_T, \\ T \models_\gamma p\!:\!A & \text{if} & \gamma(p) \Vdash_T A. \end{array}$$

We write $\Gamma \models_T \Delta$ to mean that, for all environments γ, if, for all $J \in \Gamma$, it holds that $T \models_\gamma J$ then there exists $K \in \Delta$ such that $T \models_\gamma K$. We write $\Gamma \models \Delta$ to mean that $\Gamma \models_T \Delta$ for all models T.

The sequent calculus uses *sequents* of the form $\Gamma \Longrightarrow \Delta$, where Γ and Δ are finite, which are to be read as expressing that $\Gamma \models \Delta$. As we saw in Section 1, there are problems in obtaining a cut-free system for arbitrary sequents. We avoid these problems by defining a proof system operating on a restricted class of sequents.

The restricted class of sequents is obtained by imposing conditions on the left-hand set of judgements.

A (possibly infinite) set of judgements, Γ, is said to be *assumable* if it satisfies the following three conditions.

1. If $p \xrightarrow{a} q \in \Gamma$ then q is a process variable.
2. If $p \xrightarrow{a} x \in \Gamma$ and $q \xrightarrow{b} x \in \Gamma$ then $p = q$ (syntactic identity) and $a = b$.
3. The relation, $\lhd_\Gamma$, on process variables, defined by $x \lhd_\Gamma y$ if there exists $p \xrightarrow{a} y \in \Gamma$ such that p contains x, is well-founded.

We call a sequent, $\Gamma \Longrightarrow \Delta$, *admissible* if Γ is assumable. Our sequent calculus will work with admissible sequents.

Conditions 1–3 above are the simplest we could find with which we could obtain a cut-elimination theorem. The three counterexamples from Section 1 are ruled out by conditions 1 and 2. Condition 3 prevents, for example, judgements $p \xrightarrow{a} x$ from occurring in Γ when $x \in \mathit{Vars}(p)$. For p containing arbitrary GSOS operators (involving negative premises) it may be impossible to satisfy such judgements for reasons to do with the nonexistence of solutions to arbitrary unguarded recursion equations. As in Section 1, there exist examples of such judgements for which the sequent $p \xrightarrow{a} x \Longrightarrow$ requires cut to be derivable. One final observation concerning assumability is that, if one reads $p \xrightarrow{a} q$ as a judgement that q has "type" $p \xrightarrow{a}$, then the conditions on assumability are just an infinitary generalization of the usual requirements on contexts in dependent type theory.

We now give the proof rules for the sequent calculus. Each rule is to be read as applying only when the hypotheses and conclusion are admissible sequents. As usual we have the axiom rule:

$$\frac{}{\Gamma, J \Longrightarrow J, \Delta}\ (\mathrm{Ax})$$

Admissibility considerations mean, for example, that when J is an action judgement it must have the form $p \xrightarrow{a} x$. As a matter of fact, it will follow from the completeness proof of Section 4 that (Ax) need only ever be applied with J of the form $y \xrightarrow{a} x$.

The rules for logical judgements are presented in Figure 1. These rules essentially form a sequent calculus for a multi-modality version of the minimal modal logic K, albeit with the extra baggage of process terms and (in)action judgements.

The rules for inaction judgements are presented in Figure 2. They are a straightforward implementation of the definition of $\xrightarrow{a}\!\!\!\!\!/$ in terms of $\xrightarrow{a}$.

The rules for action judgements are presented in Figure 3. They are determined by the GSOS system $\mathcal{R}$. Suppose that $\mathcal{R}$ contains exactly l rules with operator f and action c, so for each h with $1 \leq h \leq l$ we have a distinct rule:

$$\frac{\{x_i \xrightarrow{a_{hij}} y_{ij}\}_{1 \leq j \leq m_{hi}}^{1 \leq i \leq k} \quad \{x_i \xrightarrow{b_{hij}}\}_{1 \leq j \leq n_{hi}}^{1 \leq i \leq k}}{f(x_1, \ldots, x_k) \xrightarrow{c} r_h(\vec{x}, \vec{y})} \qquad (2)$$

Then we have l sequent rules introducing action judgements of the form $f(p_1, \ldots, p_k) \xrightarrow{c} r$ on the right, namely $(f\xrightarrow{c}\mathrm{R})_1$, ..., $(f\xrightarrow{c}\mathrm{R})_l$, and one rule introducing such judgements on the left, namely $(f\xrightarrow{c}\mathrm{L})$. Note that in any application of $(f\xrightarrow{c}\mathrm{L})$, when $l > 0$, it must be the case that $f(p_1, \ldots, p_k) \xrightarrow{c} x \notin \Gamma$, as otherwise the premises would not be admissible. We give examples of the rules generated by some specific process operators below.

Lastly, we consider a substitution rule and two cut rules. Although these rules will turn out to be admissible, they are useful for practical applications (see Section 6). The substitution rule is simply:

$$\frac{\Gamma \Longrightarrow \Delta}{\Gamma[p/x] \Longrightarrow \Delta[p/x]}\ (\mathrm{Sub})$$

restricted to apply only in cases that preserve admissibility. For logical judgements and inaction judgements, J, we include the usual cut rule:

$$\frac{\Gamma, J \Longrightarrow \Delta \quad \Gamma \Longrightarrow J, \Delta}{\Gamma \Longrightarrow \Delta}\ (\mathrm{Cut})$$

For action judgements there is a natural generalization of the usual rule that allows one to cut out an arbitrary action judgement from the right-hand side of a sequent. The rule is

$$\frac{\Gamma, p \xrightarrow{a} x \Longrightarrow \Delta \quad \Gamma \Longrightarrow p \xrightarrow{a} q, \Delta}{\Gamma[q/x] \Longrightarrow \Delta[q/x]}\ (\mathrm{ActCut})$$

Thus (ActCut) combines (Cut) and (Sub) in a way that is consistent with the admissibility requirements. It is worth mentioning that (ActCut) is not an arbitrary generalization of the usual cut rule. Although we shall not consider a syntactic proof of cut-elimination in this paper, (ActCut) is needed to define the reductions on derivations involved in such a proof.

We have now presented the entire system. Note that no structural rules were given. Exchange and contraction are redundant because sequents are built from finite sets. Weakening is an admissible rule.

Before stating the theorems we give some illustrative examples of the induced rules for particular process operators. For the prefix, zero and sum operators, the

$$\frac{}{\Gamma \Longrightarrow p{:}\top, \Delta}\ (\top\mathrm{R})$$

$$\frac{\Gamma \Longrightarrow p{:}A, \Delta}{\Gamma, p{:}\neg A \Longrightarrow \Delta}\ (\neg\mathrm{L}) \qquad \frac{\Gamma, p{:}A \Longrightarrow \Delta}{\Gamma \Longrightarrow p{:}\neg A, \Delta}\ (\neg\mathrm{R})$$

$$\frac{\Gamma, p{:}A, p{:}B \Longrightarrow \Delta}{\Gamma, p{:}A \wedge B \Longrightarrow \Delta}\ (\wedge\mathrm{L}) \qquad \frac{\Gamma \Longrightarrow p{:}A, \Delta \quad \Gamma \Longrightarrow p{:}B, \Delta}{\Gamma \Longrightarrow p{:}A \wedge B, \Delta}\ (\wedge\mathrm{R})$$

$$\frac{\Gamma, p \xrightarrow{a} x, x{:}A \Longrightarrow \Delta}{\Gamma, p{:}\langle a\rangle A \Longrightarrow \Delta}\ (\langle a\rangle\mathrm{L})^{*} \qquad \frac{\Gamma \Longrightarrow p \xrightarrow{a} q, \Delta \quad \Gamma \Longrightarrow q{:}A, \Delta}{\Gamma \Longrightarrow p{:}\langle a\rangle A, \Delta}\ (\langle a\rangle\mathrm{R})$$

*Restriction on ($\langle a\rangle$L): the variable x must not occur in the rule conclusion.

Figure 1: Rules for logical judgements

$$\frac{\Gamma \Longrightarrow p \xrightarrow{a} q, \Delta}{\Gamma, p \xrightarrow{a} \Longrightarrow \Delta}\ (\xrightarrow{a}\mathrm{L}) \qquad \frac{\Gamma, p \xrightarrow{a} x \Longrightarrow \Delta}{\Gamma \Longrightarrow p \xrightarrow{a}, \Delta}\ (\xrightarrow{a}\mathrm{R})^{*}$$

*Restriction on ($\xrightarrow{a}$R): the variable x must not occur in the rule conclusion.

Figure 2: Rules for inaction judgements

$$\frac{\{\Gamma \Longrightarrow p_i \xrightarrow{a_{hij}} q_{hij}, \Delta\}_{1\le j\le m_{hi}}^{1\le i\le k} \quad \{\Gamma \Longrightarrow p_i \xrightarrow{b_{hij}}, \Delta\}_{1\le j\le n_{hi}}^{1\le i\le k}}{\Gamma \Longrightarrow f(p_1,\ldots,p_k) \xrightarrow{c} r_h(\vec{p},\vec{q}), \Delta}\ (f\xrightarrow{c}\mathrm{R})_h$$

$$\frac{\left\{ \Gamma[r_h(\vec{p},\vec{y})/x], \{p_i \xrightarrow{a_{hij}} y_{ij}\}_{1\le j\le m_{hi}}^{1\le i\le k}, \{p_i \xrightarrow{b_{hij}}\}_{1\le j\le n_{hi}}^{1\le i\le k} \Longrightarrow \Delta[r_h(\vec{p},\vec{y})/x] \right\}_{1\le h\le l}}{\Gamma, f(p_1,\ldots,p_k) \xrightarrow{c} x \Longrightarrow \Delta}\ (f\xrightarrow{c}\mathrm{L})^{*}$$

*Restriction on ($f\xrightarrow{c}$L): all of the variables y_{ij} are distinct and do not occur in the rule conclusion.

Figure 3: Rules for action judgements

$$\frac{\Gamma \Longrightarrow p \xrightarrow{a} p', \Delta}{\Gamma \Longrightarrow p \rangle\!\rangle q \xrightarrow{a} p' \rangle\!\rangle q, \Delta}\ (\rangle\!\rangle\xrightarrow{a}\mathrm{R})_1 \qquad \frac{\Gamma \Longrightarrow p \xrightarrow{a}, \Delta \quad \Gamma \Longrightarrow q \xrightarrow{a} q', \Delta}{\Gamma \Longrightarrow p \rangle\!\rangle q \xrightarrow{a} p \rangle\!\rangle q', \Delta}\ (\rangle\!\rangle\xrightarrow{a}\mathrm{R})_2$$

$$\frac{\Gamma[x\rangle\!\rangle q\,/\,z], p \xrightarrow{a} x \Longrightarrow \Delta[x\rangle\!\rangle q\,/\,z] \qquad \Gamma[p\rangle\!\rangle y\,/\,z], p \xrightarrow{a}, q \xrightarrow{a} y \Longrightarrow \Delta[p\rangle\!\rangle y\,/\,z]}{\Gamma, p\rangle\!\rangle q \xrightarrow{a} z \Longrightarrow \Delta}\ (\rangle\!\rangle\xrightarrow{a}\mathrm{L})^{*}$$

*Restriction on ($\rangle\!\rangle\xrightarrow{a}$L): x and y do not occur in the rule conclusion.

Figure 4: Rules for the $\rangle\!\rangle$ operator

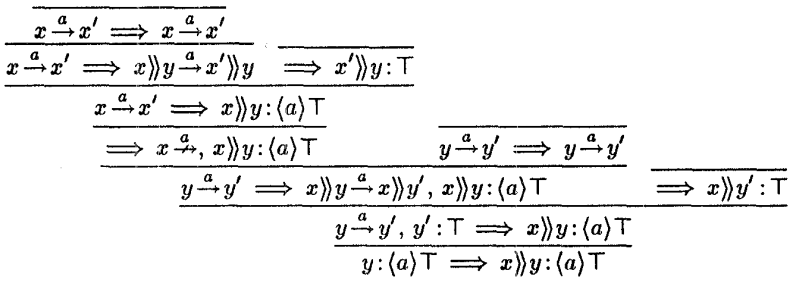

Figure 5: Example derivation in the sequent calculus

right-hand rules are the same as those given earlier in Section 1. The left-hand rules differ in that they are specifically tailored to admissible sequents. The new versions are:

$$\frac{\Gamma[p/x] \Longrightarrow \Delta[p/x]}{\Gamma,\, a.p \xrightarrow{a} x \Longrightarrow \Delta} \qquad \frac{}{\Gamma,\, a.p \xrightarrow{b} x \Longrightarrow \Delta}\ a \neq b$$

$$\frac{}{\Gamma,\, 0 \xrightarrow{a} x \Longrightarrow \Delta}$$

$$\frac{\Gamma[y/x],\, p \xrightarrow{a} y \Longrightarrow \Delta[y/x] \quad \Gamma[z/x],\, q \xrightarrow{a} z \Longrightarrow \Delta[z/x]}{\Gamma,\, p+q \xrightarrow{a} x \Longrightarrow \Delta}$$

where in the last rule y and z do not occur in the concluding sequent. Note that all the rules are special cases of their earlier counterparts. For a last example, we consider a prioritized parallel operator, $\gg$, chosen to illustrate the use of negative premises. The operational rules for $\gg$ are:

$$\frac{x \xrightarrow{a} x'}{x \gg y \xrightarrow{a} x' \gg y} \qquad \frac{x \overset{a}{\not\rightarrow} \quad y \xrightarrow{a} y'}{x \gg y \xrightarrow{a} x \gg y'}$$

Thus the right-hand argument may only perform an a action when the left-hand argument cannot. The derived sequent rules for $\gg$ are presented in Figure 4.

To show the proof system at work, we give in Figure 5 an example derivation of $y : \langle a \rangle \top \Longrightarrow x \gg y : \langle a \rangle \top$. For readability, we avoid including extraneous judgements in the sequents. The full derivation involves evident weakenings of the written sequents.

We end this section with the main results. For assumable (possibly infinite) Γ and arbitrary Δ we write $\Gamma \vdash \Delta$ to mean that there exist finite subsets $\Gamma' \subseteq \Gamma$ and $\Delta' \subseteq \Delta$ such that the sequent $\Gamma' \Longrightarrow \Delta'$ (which is necessarily admissible) is derivable. Similarly, we write $\Gamma \vdash_{cf} \Delta$ to mean that for some finite subsets $\Gamma' \subseteq \Gamma$, $\Delta' \subseteq \Delta$ the sequent $\Gamma' \Longrightarrow \Delta'$ is derivable without use of any of the rules: (Sub), (Cut) and (ActCut).

Theorem 1 (Soundness) *If* $\Gamma \vdash \Delta$ *then* $\Gamma \models \Delta$

Soundness is proved by the standard induction on derivations.

Theorem 2 (Cut-free completeness) *If* Γ *is assumable and* $\Gamma \models \Delta$ *then* $\Gamma \vdash_{cf} \Delta$.

Completeness will be proved in Section 4. The syntactic importance of the two theorems is that we have our compositional proof system:

Corollary (Cut elimination) $\Gamma \vdash \Delta$ *if and only if* $\Gamma \vdash_{cf} \Delta$.

4 Proof of completeness

This section is devoted to the proof of Theorem 2. As usual we prove the contrapositive. Suppose that $\Gamma_0 \not\vdash_{cf} \Delta_0$ where Γ_0 is assumable. We shall construct a model T_c together with an environment γ_c showing that $\Gamma_0 \not\models \Delta_0$.

A *substitution*, σ, is a partial function from variables to process terms. We write $Dom(\sigma)$ for the domain of σ. We also use set-theoretic notation for manipulating partial functions, which are considered as their graphs. We shall construct a sequence of triples $(\Gamma_i, \Delta_i, \sigma_i)$, for $i \geq 0$. For each i, we write U_i for the set of all process variables appearing in $\bigcup_{j \leq i} \Gamma_j \cup \Delta_j$ and V_i for $U_i \backslash Dom(\sigma_i)$. The sequence will satisfy the following properties:

1. Γ_i is assumable;
2. $\Gamma_i \not\vdash_{cf} \Delta_i$;
3. $\sigma_i \subseteq \sigma_{i+1}$; and

4. there are infinitely many variables not contained in U_i.

We already have Γ_0 and Δ_0. Define $\sigma_0 = \emptyset$. (It may be assumed, without loss of generality, that there are infinitely many variables not contained in U_0.)

To define the rest of the sequence let $\{\tau_0, \tau_1, \ldots\}$ be an enumeration of the following "scheduling" set:

$\{(J, \vec{q}, m) \mid J$ a judgement, $\vec{q}$ a (possibly empty) vector of process terms and $m \geq 0\}$.

such that each element of the set appears infinitely often in the enumeration. Define $\Gamma_{i+1} = \Gamma_i$ and $\Delta_{i+1} = \Delta_i$ and $\sigma_{i+1} = \sigma_i$ *unless* one of the following holds:

- $\tau_i = (p{:}\neg A, \epsilon, 0)$ and $p{:}\neg A \in \Gamma_i$, in which case $\Delta_{i+1} = \Delta_i \cup \{p{:}A\}$;
- $\tau_i = (p{:}A \wedge B, \epsilon, 0)$ and $p{:}A \wedge B \in \Gamma_i$, in which case $\Gamma_{i+1} = \Gamma_i \cup \{p{:}A, p{:}B\}$;
- $\tau_i = (p{:}\langle a\rangle A, \epsilon, 0)$ and $p{:}\langle a\rangle A \in \Gamma_i$, in which case $\Gamma_{i+1} = \Gamma_i \cup \{p \xrightarrow{a} x, x : A\}$ where x is a chosen variable not contained in U_i;
- $\tau_i = (p \xrightarrow{a}\!\!\!\!\!\!\cdot\;, q, 0)$, $p \xrightarrow{a}\!\!\!\!\!\!\cdot\; \in \Gamma_i$ and $Vars(q) \subseteq V_i$, in which case $\Delta_{i+1} = \Delta_i \cup \{p \xrightarrow{a} q\}$;
- $\tau_i = (f(p_1, \ldots, p_k) \xrightarrow{c} x, \epsilon, 0)$ and $f(p_1, \ldots, p_k) \xrightarrow{c} x \in \Gamma_i$, in which case:

$$\begin{aligned}\Gamma_{i+1} &= (\Gamma_i \backslash \{f(p_1, \ldots, p_k) \xrightarrow{c} x\})[r_h(\vec{p}, \vec{y})/x] \cup \\ &\quad \{p_i \xrightarrow{a_{hij}} y_{ij}\}_{1 \leq j \leq m_{hi}}^{1 \leq i \leq k} \cup \{p_i \xrightarrow{b_{hij}}\!\!\!\!\!\!\cdot\;\}_{1 \leq j \leq n_{hi}}^{1 \leq i \leq k} \\ \Delta_{i+1} &= \Delta_i[r_h(\vec{p}, \vec{y})/x] \\ \sigma_{i+1} &= \sigma_i \cup \{(x, r_h(\vec{p}, \vec{y}))\}\end{aligned}$$

  where $\vec{y}$ is a chosen vector of distinct variables not contained in U_i and h is chosen so $\Gamma_{i+1} \not\vdash_{cf} \Delta_{i+1}$;
- $\tau_i = (p{:}\neg A, \epsilon, 1)$ and $p{:}\neg A \in \Delta_i$, in which case $\Gamma_{i+1} = \Gamma_i \cup \{p{:}A\}$;
- $\tau_i = (p{:}A \wedge B, \epsilon, 1)$ and $p{:}A \wedge B \in \Delta_i$, in which case $\Delta_{i+1} = \Delta_i \cup \{p{:}A\}$ if $\Gamma_i \not\vdash_{cf} p{:}A$, Δ_i, and $\Delta_{i+1} = \Delta_i \cup \{p{:}B\}$ otherwise;
- $\tau_i = (p{:}\langle a\rangle A, q, 1)$, $p{:}\langle a\rangle A \in \Delta_i$ and $Vars(q) \subseteq V_i$, in which case if $\Gamma_i \not\vdash_{cf} q{:}A$, Δ_i then $\Delta_{i+1} = \Delta_i \cup \{q{:}A\}$, otherwise $\Delta_{i+1} = \Delta_i \cup \{p \xrightarrow{a} q\}$.
- $\tau_i = (p \xrightarrow{a}\!\!\!\!\!\!\cdot\;, \epsilon, 1)$ and $p \xrightarrow{a}\!\!\!\!\!\!\cdot\; \in \Delta_i$, in which case $\Gamma_{i+1} = \Gamma_i \cup \{p \xrightarrow{a} x\}$ where x is a chosen variable not contained in U_i;
- $\tau_i = (f(p_1, \ldots, p_k) \xrightarrow{c} r_h(\vec{p}, \vec{q}), \vec{q}, h)$, $Vars(\vec{q}) \subseteq V_i$ and $f(p_1, \ldots, p_k) \xrightarrow{c} r_h(\vec{p}, \vec{q}) \in \Delta_i$, in which case: if there exist i, j with $1 \leq i \leq k$ and $1 \leq j \leq m_{hi}$ such that $\Gamma_i \not\vdash_{cf} p_i \xrightarrow{a_{hij}} q_{ij}$, Δ_i then $\Delta_{i+1} = \Delta_i \cup \{p_i \xrightarrow{a_{hij}} q_{ij}\}$ for a chosen such i, j; otherwise $\Delta_{i+1} = \Delta \cup \{p_i \xrightarrow{b_{hij}}\!\!\!\!\!\!\cdot\;\}$ for a chosen i, j with $1 \leq i \leq k$ and $1 \leq j \leq n_{hi}$ such that $\Gamma_i \not\vdash_{cf} p_i \xrightarrow{b_{hij}}\!\!\!\!\!\!\cdot\;$, Δ_i;

where we write ϵ for the empty vector and, in the action judgement cases, it is assumed that the rules in $\mathcal{R}$ with operator f and action c have the form in (2) of Section 3, and that the vectors $\vec{y}$ and $\vec{q}$ have the appropriate length. (Similar assumptions are made below without further comment.)

Lemma 1 *The sequence $(\Gamma_i, \Delta_i, \sigma_i)$ is well defined and enjoys properties 1-4 above.*

The proof is a routine verification.

We now define the required model T_c. We write σ_ω for $\bigcup_i \sigma_i$ and U_ω for $\bigcup_i U_i$. Define $V_\omega = U_\omega \backslash Dom(\sigma_\omega)$. The model T_c is determined as the unique model such that: $|T_c| = \{p \mid Vars(p) \subseteq V_\omega\}$; the operators are interpreted using the term algebra structure; and $x \xrightarrow{a}_{T_C} q$ holds if and only if, for almost all j (i.e. for all but finitely many j), it holds that $x \xrightarrow{a} q \in \Gamma_j$ (in which case q must be a process variable). The existence and uniqueness of T_c corresponds to the analogous result for GSOS systems with δ-rules in [3].

We shall define the required environment using an iterated substitution. For a substitution σ, we write σ^* for the homomorphism on process terms satisfying:

$$\sigma^*(x) = \begin{cases} \sigma^*(\sigma(x)) & \text{if } x \in Dom(\sigma), \\ x & \text{otherwise,} \end{cases}$$

when a unique such homomorphism exists. We write $\sigma^*(J)$ for the judgement obtained by substituting $\sigma^*(x)$ for all occurrences of each variable x in J.

Lemma 2 *For all i, we have that σ_i^* exists and $x \in U_i$ implies $Vars(\sigma_i^*(x)) \in V_i$. Moreover if $J \in \Gamma_i$ then, for all $j \geq i$, either $\sigma_j^*(J) \in \Gamma_j$ or J has the form $p \xrightarrow{a} x$ where $x \in Dom(\sigma_j)$. Similarly, if $J \in \Delta_i$ then, for all $j \geq i$, it holds that $\sigma_j^*(J) \in \Delta_j$.*

Lemma 3 *The function σ_ω^* exists and $x \in U_\omega$ implies $Vars(\sigma_\omega^*(x)) \subseteq V_\omega$. Moreover, for any p, it holds that $\sigma_\omega^*(p) = \sigma_j^*(p)$ for almost all j.*

The proof of Lemma 2 is straightforward. The proof of Lemma 3 relies on the well-foundedness of the $\lhd_{\Gamma_i}$ relations. The (slightly technical) argument is omitted for lack of space.

Define $\gamma_c(x)$ to be $\sigma^*_\omega(x)$ if, $x \in U_\omega$ and arbitrary otherwise.

Lemma 4 *For all i and judgements J,*

1. *if $J \in \Gamma_i$ then $T_c \models_{\gamma_c} J$; and*
2. *if $J \in \Delta_i$ then $T_c \not\models_{\gamma_c} J$.*

Proof. First the lemma is proved for judgements $p \xrightarrow{c} q$ and $p \xrightarrow{c}$, by induction on the structure of $\sigma^*_\omega(p)$. For action judgements there are two cases:

$p \xrightarrow{c} x \in \Gamma_i$. If $x \notin Dom(\sigma_\omega)$ then $\sigma^*_\omega(p) \xrightarrow{c} x \in \Gamma_j$ for almost all $j \geq i$. But then $\sigma^*_\omega(p)$ must be a variable y, as otherwise for some τ_j of the form $(\sigma^*_\omega(p) \xrightarrow{c} x, \epsilon, 0)$ we would have $x \in Dom(\sigma_{j+1})$, a contradiction. So $y \xrightarrow{c}_{T_c} x$. Thus $T_c \models_{\gamma_c} p \xrightarrow{c} x$. Otherwise, if $x \in Dom(\sigma_\omega)$ then for some $j \geq i$, $x \in Dom(\sigma_{j+1}) \backslash Dom(\sigma_j)$. Thus $\sigma^*_j(p) \xrightarrow{c} x \in \Gamma_j$ and $\tau_j = (\sigma^*_j(p) \xrightarrow{c} x, \epsilon, 0)$ where $\sigma^*_j(p)$ has the form $f(p_1, \ldots, p_k)$. So, for some h, we have

$$\{p_i \xrightarrow{a_{hij}} y_{ij}\}^{1 \leq i \leq k}_{1 \leq j \leq m_{hi}} \cup \{p_i \overset{b_{hij}}{\nrightarrow}\}^{1 \leq i \leq k}_{1 \leq j \leq n_{hi}} \subseteq \Gamma_{j+1}.$$

Now $\sigma^*_\omega(p) = f(\sigma^*_\omega(p_1), \ldots, \sigma^*_\omega(p_k))$. So, by the induction hypothesis, $T_c \models_{\gamma_c} p_i \xrightarrow{a_{hij}} y_{ij}$ and $T_c \models_{\gamma_c} p_i \overset{b_{hij}}{\nrightarrow}$ (for all appropriate i, j). But then $T_c \models_{\gamma_c} f(p_1, \ldots, p_k) \xrightarrow{c} r_h(\vec{p}, \vec{y})$. Also $\sigma^*_\omega(x) = \sigma^*_\omega(r_h(\vec{p}, \vec{y}))$. Thus indeed $T_c \models_{\gamma_c} p \xrightarrow{c} x$.

$p \xrightarrow{c} r \in \Delta_i$. Suppose, for contradiction, that $T_c \models_{\gamma_c} p \xrightarrow{c} r$.

If $\sigma^*_\omega(p)$ is a variable x then $x \xrightarrow{c}_{T_c} \sigma^*_\omega(r)$. Thus $\sigma^*_\omega(r)$ is a variable y and $x \xrightarrow{c} y \in \Gamma_j$ for almost all j. But, for almost all j, we have $\sigma^*_j(p) = x$ and $\sigma^*_j(r) = y$, so $x \xrightarrow{c} y \in \Delta_j$. Thus $\Gamma_j \vdash_{cf} \Delta_j$, a contradiction.

If $\sigma^*_\omega(p) = f(p_1, \ldots, p_k)$ then $f(p_1, \ldots, p_k) \xrightarrow{c}_{T_c} \sigma^*_\omega(r)$. So there exist h and $\vec{q}$ such that $\sigma^*_\omega(r) = r_h(\vec{p}, \vec{q})$ and:

1. for all i, j with $1 \leq i \leq k$ and $1 \leq j \leq m_{hi}$, $p_i \xrightarrow{a_{hij}}_{T_c} q_{ij}$; and
2. for all i, j with $1 \leq i \leq k$ and $1 \leq j \leq n_{hi}$, $p_i \overset{b_{hij}}{\nrightarrow}_{T_c}$.

For almost all j, we have $\sigma^*_\omega(p) \xrightarrow{c} \sigma^*_\omega(r) \in \Delta_j$. So for some $\tau_j = (\sigma^*_\omega(p) \xrightarrow{c} \sigma^*_\omega(r), \vec{q}, h)$ it holds that either $p_i \xrightarrow{a_{hij}} q_{ij} \in \Delta_{j+1}$ or $p_i \overset{b_{hij}}{\nrightarrow} \in \Delta_{j+1}$ for some suitable i, j. Then, by the induction hypothesis, either $T_c \not\models_{\gamma_c} p_i \xrightarrow{a_{hij}} q_{ij}$ or $T_c \not\models_{\gamma_c} p_i \overset{b_{hij}}{\nrightarrow}$. Thus either way contradicts 1 or 2 above.

The cases for inaction judgements follow fairly easily.

For logical judgements, $p : A$, the lemma is proved by induction on the structure of A. We consider only the cases for the modality.

$p : \langle a \rangle A \in \Gamma_i$. For almost all j we have $\sigma^*_\omega(p) : \langle a \rangle A \in \Gamma_j$. Thus for some $\tau_j = (\sigma^*_\omega(p) : \langle a \rangle A, \epsilon, 0)$ we have $\{\sigma^*_\omega(p) \xrightarrow{a} x, x : A\} \subseteq \Gamma_{j+1}$. Then $T_c \models_{\gamma_c} \sigma^*_\omega(p) \xrightarrow{a} x$ and $T_c \models_{\gamma_c} x : A$, the latter by the induction hypothesis. So $T_c \models_{\gamma_c} p : \langle a \rangle A$.

$p : \langle a \rangle A \in \Delta_i$. For almost all j we have that $\sigma^*_\omega(p) : \langle a \rangle A \in \Delta_j$. Let q be any element of $|T_c|$. For almost all j, $Vars(q) \subseteq V_j$. Thus for some τ_j of the form $(\sigma^*_\omega(p) : \langle a \rangle A, q, 1)$ we have that either $q : A \in \Delta_{j+1}$ or $\sigma^*_\omega(p) \xrightarrow{a} q \in \Delta_{j+1}$. In the first case, by the induction hypothesis, $T_c \not\models_{\gamma_c} q : A$. In the second case $T_c \not\models_{\gamma_c} \sigma^*_\omega(p) \xrightarrow{a} q$. So indeed $T_c \not\models_{\gamma_c} p : \langle a \rangle A$.

⊠

Theorem 2 follows.

5 The intended model

The completeness theorem is relative to entailment over the class of all models of $\mathcal{R}$. Usually one is interested in truth in the intended model $T_{\mathcal{R}}$. In this section we give conditions under which completeness does indeed hold relative to $T_{\mathcal{R}}$.

The first such completeness theorem is motivated by the observation that, in any model, the state interpreting a closed process p is bisimilar to the state p in $T_{\mathcal{R}}$. As we shall see, the proof system is complete for deriving the truth in $T_{\mathcal{R}}$ of sequents containing only closed process terms. Actually, a stronger result holds — it is enough that every process variable in a sequent is forced to represent a state interpreting a closed process. A simple syntactic condition guarantees that this is the case. We say that a pair of sets of judgements, (Γ, Δ), is *closed-generated* if Γ is assumable and every process variable, x, in $\Gamma \cup \Delta$ appears in a judgement of the form $p \xrightarrow{a} x \in \Gamma$. This condition combines with the well-foundedness of $\lhd_\Gamma$ to ensure that each minimal variable x under $\lhd_\Gamma$ appears in a (necessarily unique) judgement of the form $p \xrightarrow{a} x \in \Gamma$ where p is closed.

Theorem 3 *For closed-generated (Γ, Δ), it holds that $\Gamma \models_{T_{\mathcal{R}}} \Delta$ implies $\Gamma \vdash \Delta$.*

The theorem is a direct consequence of the above proof of Theorem 2. When (Γ_0, Δ_0) is closed-generated and

$\Gamma_0 \not\vdash \Delta_0$ it turns out that the model T_c constructed in Section 4 is itself $T_{\mathcal{R}}$. To show this, one establishes that $U_\omega = Dom(\sigma_\omega)$ (hence $V_\omega = \emptyset$). This follows from the (omitted) techniques used in the proof of Lemma 3.

Given Theorems 1 and 2, an equivalent statement to Theorem 3 is that, when (Γ, Δ) is closed-generated, then $\Gamma \models_{T_{\mathcal{R}}} \Delta$ implies $\Gamma \models \Delta$. It is interesting to note that conditions 1 and 2 on the assumability of Γ are essential for this implication to hold. For example, we have that $a.0 \xrightarrow{a} 0 + 0 \models_{T_{\mathcal{R}}}$ but not that $a.0 \xrightarrow{a} 0 + 0 \models$, as 0 and $0 + 0$ have the same denotation in the model obtained by quotienting $T_{\mathcal{R}}$ by bisimilarity.

The restriction to closed-generated consequences does not fully exploit the expressivity of sequents containing open terms. One would like a more general completeness result for sequents in which the variables need not derive from closed processes. What we seek is a form of *ω-completeness*, i.e. completeness relative to all environments interpreting process variables as closed processes in $T_{\mathcal{R}}$. In order to obtain such a result, it is necessary to make some mild expressivity assumptions on the GSOS system $\mathcal{R}$.

For a labelled transition system T, the relation $\rightarrow_T$ is defined by $s \rightarrow_T t$ if there exists a such that $s \xrightarrow{a}_T t$. We write $\rightarrow_T^+$ for the transitive closure of $\rightarrow_T$. A *finite behaviour* is a state s in a labelled transition system T such that: the set $S = \{s\} \cup \{t \mid s \rightarrow_T^+ t\}$ is finite; and the restriction of $\rightarrow_T^+$ to S is irreflexive. An important property of finite behaviours (cf. [5]) is that, for any such s in T, there exists a formula A (the *characteristic* formula of s) such that, for any state t in any transition system T', we have that t is bisimilar to s if and only if $t \Vdash_{T'} A$ (the existence of A relies on the set of all actions being finite).

The GSOS system $\mathcal{R}$ is said to *represent* a finite behaviour s in T if there exists a closed term p such that p in $T_{\mathcal{R}}$ is bisimilar to s in T. Suppose that there is some finite behaviour s in T that is not represented by $\mathcal{R}$. Let A be the characteristic formula of s. Then we have that $x : A \models_{T_{\mathcal{R}}}$, but not $x : A \vdash$. Thus ω-completeness fails. Therefore, a necessary condition for ω-completeness is that $\mathcal{R}$ represent every finite behaviour. This turns out to be also a sufficient condition for a useful class of consequences.

Theorem 4 (ω-completeness) *Suppose that $\mathcal{R}$ represents every finite behaviour. Then, for finite Γ, Δ such that Γ is assumable and Δ contains no action judgements, $\Gamma \models_{T_{\mathcal{R}}} \Delta$ implies $\Gamma \vdash \Delta$.*

The condition that $\mathcal{R}$ represent every finite behaviour is rather mild. For example, it is satisfied by any process algebra containing the prefix, zero and sum operators. The restrictions on the form of consequence are all necessary. The finiteness restrictions on Γ and Δ are required because the consequence relation $\vdash$ is compact, whereas $\models_{T_{\mathcal{R}}}$ need not be. For example, take $\mathcal{R}$ to be the GSOS containing just the prefix, zero and sum operators. Then it holds that $\models_{T_{\mathcal{R}}} \{x : [a]^n \bot \mid n \geq 0\}$, but it is clear that $\not\vdash \{x : [a]^n \bot \mid n \geq 0\}$. For an example showing why Δ is required to contain no action judgement, note that it possible to construct a GSOS system, containing the prefix and zero operators, that represents every finite behaviour and in which the only closed process term bisimilar to the zero process is 0 itself (so there is necessarily no sum operator). If $\mathcal{R}$ is such a system then $\{x : [a]\bot \mid a \text{ an action}\} \models_{T_{\mathcal{R}}} a.0 \xrightarrow{a} x$, but the corresponding sequent is not provable.

Theorem 4 is proved by establishing that, under the conditions of the theorem, $\Gamma \not\models \Delta$ implies $\Gamma \not\models_{T_{\mathcal{R}}} \Delta$. Suppose then that we have T and γ such that, for all $J \in \Gamma$, $T \models_\gamma J$ and, for all $K \in \Delta$, $T \not\models_\gamma K$. We must define a $T_{\mathcal{R}}$-environment γ' such that, for all $J \in \Gamma$, $T_{\mathcal{R}} \models_{\gamma'} J$ and, for all $K \in \Delta$, $T_{\mathcal{R}} \not\models_{\gamma'} K$.

For any transition systems S and T, the relation $\sim_m$ (the m-th approximation to bisimilarity) between $|S|$ and $|T|$ is defined by: $s \sim_0 t$ always holds; and $s \sim_{m+1} t$ holds if:

1. whenever $s \xrightarrow{a}_S s'$ there exists t' such that $t \xrightarrow{a}_T t'$ and $s' \sim_m t'$; and
2. whenever $t \xrightarrow{a}_T t'$ there exists s' such that $s \xrightarrow{a}_S s'$ and $s' \sim_m t'$.

The *modal depth*, $md(A)$, of a formula A is defined to be the maximal nesting depth of $\langle a \rangle$ operators in A. We shall need the following facts, whose proofs are not difficult.

Proposition 1 (cf. [6]) *The following are equivalent:*

1. $s \sim_m t$.
2. *For all A with $md(A) \leq m$, it holds that $s \Vdash_S A$ if and only if $t \Vdash_T A$.*

Proposition 2 *If S and T are models of $\mathcal{R}$ then, for any k-ary operator f, for any $s_1, \ldots s_k \in |S|$ and any $t_1, \ldots, t_k \in |T|$, if $s_1 \sim_m t_1$ and ... and $s_k \sim_m t_k$ then $f_S(s_1, \ldots, s_k) \sim_m f_T(t_1, \ldots, t_k)$.*

Proposition 3 *If $\mathcal{R}$ represents every finite behaviour then, for every $t \in |T|$, there exists $q \in |T_{\mathcal{R}}|$ such that $q \sim_m t$.*

We shall define γ' so that for each x it will hold that $\gamma'(x) \sim_m \gamma(x)$ for some m depending on x. To determine m we assign a *depth*, $d(p)$, to each process term p by:

$$\begin{aligned} d(x) &= \begin{cases} d(p)+1 & \text{if } p \xrightarrow{a} x \in \Gamma, \\ 0 & \text{otherwise,} \end{cases} \\ d(f(p_1,\ldots,p_k)) &= max\{d(p_1),\ldots,d(p_k)\}. \end{aligned}$$

It follows from the well-foundedness of $\lhd_\Gamma$ that $d(p)$ is well-defined. Define

$$\begin{aligned} n = max(&\{d(p)+1 \mid p \xrightarrow{a} x \in \Gamma \text{ or } p \xrightarrow{a} \in \Gamma \cup \Delta\} \cup \\ &\{d(p) + md(A) \mid p{:}A \in \Gamma \cup \Delta\}), \end{aligned}$$

using the finiteness of Γ and Δ. Clearly, for any p, we have $n \geq d(p)$.

Lemma 5 *There exists a $T_\mathcal{R}$-environment γ' such that:*

1. $\gamma'(x) \sim_{n-d(x)} \gamma(x)$; *and*
2. *if* $p \xrightarrow{a} x \in \Gamma$ *then* $\gamma'(p) \xrightarrow{a}_{T_\mathcal{R}} \gamma'(x)$.

Proof. We show that $\gamma'(x)$ can be defined so that 1 and 2 hold on the assumption that $\gamma'(y)$ is so-defined for all variables y with $d(y) < d(x)$.

When $d(x) = 0$ we use Proposition 3, setting $\gamma'(x)$ to be the q given by $t = \gamma(x)$.

When $d(x) = i+1$ we have $p \xrightarrow{a} x \in \Gamma$ for some p, so $\gamma(p) \xrightarrow{a}_T \gamma(x)$. But $d(p) = i$ so, by the assumption and Proposition 2, we have $\gamma'(p) \sim_{n-i} \gamma(p)$, Therefore, by the definition of $\sim_{n-i}$, there exists $q \in |T_\mathcal{R}|$ such that $\gamma'(p) \xrightarrow{a}_{T_\mathcal{R}} q$ and $q \sim_{n-(i+1)} \gamma(x)$. Thus we define $\gamma'(x) = q$. ⊠

Lemma 6 *For all $J \in \Gamma$, $T_\mathcal{R} \models_{\gamma'} J$ and, for all $K \in \Delta$, $T_\mathcal{R} \not\models_{\gamma'} K$.*

Proof. Immediate from Lemma 5, Propositions 1 and 2 and the definition of n. ⊠

Theorem 4 follows.

6 Discussion

We have claimed that the cut-elimination theorem gives us a "compositional" proof system. The form of compositionality obtained is that given by the structure-building nature of the proof rules. For example, in the rule ($\langle\!\rangle \xrightarrow{a}$L) of Figure 4, the $p \langle\!\rangle q \xrightarrow{a} z$ judgement in the conclusion is built up from judgements $p \xrightarrow{a} x$ and $q \xrightarrow{a} y$ in the premises. Thus the underlying principle is the compositional one of reasoning about the whole by reasoning about its parts. Note that the form of compositionality obtained applies equally to the structure of processes and to the structure of formulae.

On the other hand, another important aspect of compositionality, the *modularity* of process verification, is not addressed by the cut-free system. For example, a modular verification that $p \langle\!\rangle q$ satisfies C would involve verifying an appropriate property A of p and an appropriate property B of q, where A and B are arbitrarily complex formulae chosen so as to be sufficient to establish the desired goal. Our proof system does naturally support such a form of verification, but ironically the cut rule is crucial to this. For example, one can combine the (Sub) and (Cut) rules to obtain the following derived rule:

$$\frac{\Longrightarrow p{:}A \quad \Longrightarrow q{:}B \quad x{:}A, y{:}B \Longrightarrow x \langle\!\rangle y{:}C}{\Longrightarrow p \langle\!\rangle q{:}C}$$

which produces the desired subgoals, together with a proof obligation to justify the choice of A and B. This approach to modular verification is that adopted by Stirling in [9], who used special sequents for stating properties $x : A, y : B \Longrightarrow x||y : C$ (where $||$ is the CCS parallel operator). In our approach, such sequents arise in a uniform way and are available for all the process operators in the language. Moreover, a crucial improvement on [9] is our Theorem 4, which shows that our proof system is complete for establishing such properties.

Thus, despite cut-elimination, the cut rules will be useful in any practical implementation of the proof system. This is no surprise. In standard sequent calculi cut is an indispensible proof rule, allowing the reuse of established lemmas and a general shortening of proofs. Nevertheless, cut-free proofs are important too. For example, the structural constraints on cut-free proofs are particularly useful for guiding goal-directed proof search.

An important pragmatic issue is the ease-of-use of the proof system. The cut-elimination theorem gives some mathematical evidence for the naturality of the proof rules presented in this paper. Moreover, all the rules have one very desirable feature: each expresses a fundamental, self-explanatory property of its associated connective, modality or operator. This feature makes it plausible that natural informal proofs that a program satisfies a property (whose primitive steps should all be similarly self-explanatory) might have close formal analogues.

7 Conclusions and further work

Previous work on compositional proof systems for process algebras (see, e.g., [10, 9, 2]) has often involved ingenious ideas that work for the operators under consideration but do not easily generalize to other operators. Through having a sufficiently expressive form of sequent and incorporating the operational semantics into the proof rules, we have obtained a generic system applicable to a wide class of operators. We have also improved on previous work by allowing open process terms and proving a corresponding ω-completeness theorem.

Regarding improvements to our work, there are several limitations inherent in our use of GSOS systems. One is the restriction to a finite set of actions. There are natural generalizations to infinite action sets which, however, involve the use of infinitary rules. It would be interesting to develop a natural class of finitary rules for dealing with infinite action sets. A further limitation is that we have not included a recursion operator in the GSOS system. As remarked in [1] any process defined by guarded recursion can be dealt with by including a new process constant for the process and giving it explicit operational rules. However, it would be better to include direct proof rules for guarded recursion in the sequent calculus. Although such an extension of the proof system is not difficult, it leads to tedious technical complications in the definitions and proofs.

A severe practical limitation of our work is the use of Hennessy-Milner logic, which is too weak to express many interesting properties. It would be of great interest to investigate adding sequent rules for more powerful logical constructs, such as the least and greatest fixed-points of the modal μ-calculus [7]. With such an expressive logic, one could not hope for a completeness result for arbitrary processes. However, as in [2], it ought to be possible to obtain completeness for finite state processes. In this setting, cut-elimination is likely to be a difficult problem.

More generally, the idea of deriving Gentzen-style rules from operational semantics seems likely to have applications in other computational settings. It would be interesting to investigate this possibility by considering other programming languages (such as an imperative language) and other programming logics (such as dynamic logic).

Acknowledgements

I have benefited from discussions with Peter Sewell and Colin Stirling. This research was carried out under an EPSRC postdoctoral research fellowship.

References

[1] L. Aceto, B. Bloom, and F. Vaandrager. Turning SOS rules into equations. *Information and Computation*, 111:1–52, 1994.

[2] H. R. Anderson, C. P. Stirling, and G. Winskel. A compositional proof system for the modal μ-calculus. In *Proceedings of 9th Annual Symposium on Logic in Computer Science*, pages 144–153, 1994.

[3] B. Bloom, S. Istrail, and A. R. Meyer. Bisimulation can't be traced: preliminary report. In *Conference record of the 15th ACM Symposium on Principles of Programming Languages*, pages 229–239, 1988.

[4] G. Gentzen. Investigations into logical deduction. 1935. In M.E. Szabo, editor, *The collected papers of Gerhard Gentzen*, pages 68–128. North Holland Publishing Company, 1969.

[5] S. Graf and J. Sifakis. A modal characterization of observational congruence on finite terms of CCS. *Information and Control*, 68:125–145, 1986.

[6] M. Hennessy and R. Milner. Algebraic laws for nondeterminism and concurrency. *J. Assoc. Comput. Mach.*, 32:137–161, 1985.

[7] D. Kozen. Results on the propositional mu-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[8] R. Milner. *Communication and Concurrency*. Prentice Hall international series in computer science. Prentice Hall, 1989.

[9] C. P. Stirling. Modal logics for communicating systems. *Theoretical Computer Science*, 49:311–347, 1987.

[10] G. Winskel. A complete proof system for SCCS with modal assertions. *Fundamenta Informaticae*, IX:401–420, 1986.